

```

%!
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%   SoftTTY.ps
%
%   PostScript terminal emulator.
%   Copyright (C) 1987.
%   By Don Hopkins for Wedge Computer, Inc.
%   All rights reserved.
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%   SoftTTY is a product of Don Hopkins, and Wedge Computer, Inc. and is
%   provided for unrestricted use provided that this legend is included on
%   all tape media and as a part of the software program in whole or part.
%   Users may copy or modify SoftTTY without charge, but are not
%   authorized to license or distribute it to anyone else except as part
%   of a product or program developed by the user.
%
%   SoftTTY is provided as is with no warranties of any kind including
%   the warranties of design, merchantability and fitness for a
%   particular purpose.
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```
systemdict begin
```

```
/TTY DefaultWindow
dictbegin
```

```

% instance variables
/Rows 24 def
/Columns 80 def
/Lines null def
/CurX 0 def
/CurY 0 def
/SaveCurX 0 def
/SaveCurY 0 def
/CharWidth null def
/CharHeight null def
/CharMatrix null def
/PixelMatrix null def
/KeyProcess null def
/TTYFont null def
/LocalEcho false def
/OutputMonitor null def
/OutputProcess null def
/infile null def
/args null def
/ScrollMode true def
/WrapMode true def
/WrapBack? true def
/WrapForward? true def
/Auto-CR false def
/cursor-can null def
/cursor-delay 1.5 60 div def
/cursor-state false def
/last-x 0 def
/last-y 0 def

```

```

/error-count null def
/BottomMargin 24 def
/TopMargin 0 def
dictend
classbegin
% class variables
% /SavedSize 258 def
/TextMargin 6 def
/TextVSpace 5 def
/TextVOffset 0 def
/ForkPaintClient false def
/FontSize 12 def
/FontName /Screen def
/IconImage /gern def
/BellTime .2 60 div def
/Spaces 512 string def
0 1 511 {Spaces exch 32 put} for

% methods
/new {
/new super send
dup begin
/PaintClient {
OutputMonitor { refresh } monitor
} def
/OutputMonitor createmonitor def
end
} def

/CreateClientCanvas {
/CreateClientCanvas super send
/fontmenu
[ (Screen) (Screen-Bold)
(Courier) (Courier-Bold) (Courier-Oblique) (Courier-BoldOblique)
(Times-Roman) (Times-Bold) (Times-Italic) (Times-BoldItalic)
(Helvetica) (Helvetica-Bold) (Helvetica-BoldOblique)
(Boston) (Cyrillic) (Symbol) (Icon) (ExampleFont) ]
[{currentkey cvn /setfontname myterm send}] % XXX Depends on psexec !!!
/new DefaultMenu send def
/pointsize menu
[(4) (6) (8) (10) (12) (14) (16) (18) (24) (32) (64) (72) (92)]
[{currentkey cvi /setfontsize myterm send}]
/new DefaultMenu send def
/ClientMenu
[ (Fonts) fontmenu
(No Echo) {false /setlocalecho myterm send}
(Points) pointsize menu
(Echo) {true /setlocalecho myterm send} ]
/new DefaultMenu send def
} def

/setlocalecho { % bool =>
/LocalEcho exch def
} def

/setfontnamesize { % name size => -
/FontSize exch def
/FontName exch def

```

```

    new-font fit-font new-cursor paintclient
} def

/setfontsize { % size => -
    FontName exch setfontnamesize
} def

/setfontname { % name => -
    FontSize setfontnamesize
} def

/new-font {
    false setprintermatch
    /TTYFont FontName findfont FontSize scalefont def
    framebuffer setcanvas
    TTYFont setfont
    (gy_MTWY()/|{ }) stringbbox
    /TextVOffset TextVSpace 4 index sub def
    exch pop exch sub
    TextVSpace add /CharHeight exch def
    pop
    (x) stringwidth pop 1 max
    /CharWidth exch def
    /PixelMatrix matrix currentmatrix def
    initmatrix
    TextMargin CharHeight TextMargin add translate
    CharWidth CharHeight scale
    /CharMatrix matrix currentmatrix def
    ClientCanvas null ne { new-cursor } if
} def

/new-cursor {
    gsave
    ClientCanvas setcanvas
    cursor-can null eq {
        /cursor-can ClientCanvas newcanvas def
        cursor-can /Transparent false put
        cursor-can /Retained true put
    } if
    shape-cursor
    grestore
} def

/reshape {
    new-font
    /reshape super send
    ClientCanvas /Transparent false put
    ClientCanvas /Retained true put
    new-cursor
    fit-font
} def

/preshape { % x y => -
    new-font
    fit-window
    new-cursor
} def

```

```

/fit-window {
  gsave
    TextMargin dup add Rows CharWidth mul add
    TextMargin dup add Columns CharHeight mul add
    /reshape super send
    new-cursor
    reshape-lines
  grestore
} def

/fit-font {
  gsave
    ClientCanvas setcanvas
    clippath pathbbox points2rect
    TextMargin dup add sub CharHeight div floor 1 max /Rows exch def
    TextMargin dup add sub CharWidth div floor 1 max /Columns exch def
    pop pop
    reshape-lines
  grestore
} def

/reshape-lines {
%   /SavedLines SavedSize array def
  /OldLines Lines def
  /Lines [
    Rows {
      Columns string
      dup 0 Spaces 0 Columns getinterval putinterval
    } repeat
  ] def
  OldLines null ne {
    0 1 OldLines length Rows min 1 sub { /i exch def
      Lines i get 0
      OldLines i get
      dup length Columns gt {
        0 Columns getinterval
      } if
      putinterval
    } for
  } if
  /OldLines null def
  /TopMargin 0 def
  /BottomMargin Rows def
  /CurX CurX Columns 1 sub min def
  /CurY CurY Rows 1 sub min def
  update-cursor
} def

/refresh {
  ClientCanvas setcanvas
  clippath 1 setgray fill
  Rows 0 ne {
    0 1 Rows 1 sub {
      pause
      put-line } for
  } if
} def

```

```

/slow-refresh {
  ClientCanvas setcanvas
  Rows 0 ne {
    0 1 Rows 1 sub {
      wipe-line
      put-line } for
  } if
} def

% Terminal methods

/char-pos { % col row => x y
  gsave
  CharMatrix setmatrix
  transform
  grestore
%   CharMatrix transform
} def

/wipe-line { % line => line
  newpath
  Columns 0 2 index
  char-rect
  1 setgray fill
} def

/put-line { % line => -
  0 1 index char-pos
  TextVOffset add moveto
  Lines exch get
  TTYFont setfont
  0 setgray show
} def

/put-string { % str => -
  {
    dup length CurX add Columns lt { exit } if
    Columns CurX sub
    2 copy 0 exch getinterval put-string-that-fits
    2 copy exch length exch sub getinterval
    pause
  } loop
  dup length 0 ne { put-string-that-fits } { pop } ifelse
} def

/put-string-that-fits { % str => -
  dup length 0 ne {
    Lines CurY get CurX 2 index putinterval
    ClientCanvas setcanvas
    newpath
    dup length CurX CurY char-rect
    1 setgray fill
    CurX CurY char-pos
    TextVOffset add moveto
    TTYFont setfont
    0 setgray dup show
    length move-cur-x
  } if

```

```

} def

/set-cur { % x y => -
  Rows 1 sub min 0 max /CurY exch def
  Columns 1 sub min 0 max /CurX exch def
} def

/home-cur {
  0 0 set-cur
} def

/move-cur-x { % delta => -
  CurX add
  dup 0 lt {
    WrapBack? {
      Columns 1 sub 1 index sub Columns div floor
      dup line-starves
      Columns mul add
    } {
      pop 0
    } ifelse
  } {
    dup Columns ge {
      WrapForward? {
        dup Columns div floor
        dup line-feeds
        Columns mul sub
      } {
        pop Columns 1 sub
      } ifelse
    } if
  } ifelse
  /CurX exch def
} def

/line-feeds { % count => -
  CurY add
  dup BottomMargin ge {
    CurY BottomMargin ge {
      Rows 1 sub min
    } {
      BottomMargin 1 sub sub
      TopMargin BottomMargin 3 -1 roll scroll-region
      BottomMargin 1 sub
    } ifelse
  } if
  /CurY exch def
} def

/line-starves { % count => -
  CurY exch sub
  dup TopMargin lt {
    CurY TopMargin lt {
      0 max
    } {
      TopMargin sub
      TopMargin BottomMargin 3 -1 roll scroll-region
      TopMargin
    } if
  }

```

```

    } ifelse
  } if
  /CurY exch def
} def

/old-move-cur-y { % delta => -
  ScrollMode {
    CurY add
    0 max
    dup BottomMargin ge {
      BottomMargin sub 1 add
      0 exch delete-lines
      BottomMargin 1 sub
    } if
  } {
    dup 0 le {
      CurY add
      % I don't trust mod with negative numbers one bit.
      { dup 0 ge {exit} if
        Rows add } loop
    } {
      CurY
      exch {
        1 add
        dup Rows ge { Rows sub } if
        dup erase-line
      } repeat
    } ifelse
  } ifelse
  /CurY exch def
} def

/delete-lines { % first #lines => -
  Rows exch
  scroll-region
} def

/insert-lines { % first #lines => -
  neg Rows exch
  scroll-region
} def

/delete-chars { % #chars => -
  dup 0 ne {
    /char-count exch def
    /after-cur Columns CurX sub def
    /to-copy after-cur char-count sub 0 max def
    Lines CurY get
    to-copy 0 ne {
      dup dup
      to-copy CurX char-count add
      2 copy CurY char-rect
      char-count CharWidth mul neg 0 copyarea
      exch getinterval
      CurX exch putinterval
    } if
    after-cur to-copy sub
    CurX to-copy add
  }

```

```

    2 copy CurY char-rect
    1 setgray fill
    Spaces 0 4 -1 roll getinterval putinterval
  } {pop} ifelse
} def

/insert-chars { % #chars => -
  dup 0 ne {
    /char-count exch def
    /after-cur Columns CurX sub def
    /to-copy after-cur char-count sub 0 max def
    Lines CurY get
    to-copy 0 ne {
      dup dup
      to-copy CurX CurY char-rect
      char-count CharWidth mul 0 copyarea
      CurX to-copy getinterval
      CurX char-count add exch putinterval
    } if
    after-cur to-copy sub
    dup CurX CurY char-rect
    1 setgray fill
    CurX Spaces 0 4 -1 roll getinterval putinterval
  } {pop} ifelse
} def

/update-line { % line => -
  newpath
  Columns 0 2 index char-rect
  1 setgray fill
  put-line
} def

/char-rect { % width col row => -
  char-pos
  3 -1 roll CharWidth mul CharHeight
  rectpath
} def

/flash-cursor {
  animate-cursor
  cursor-can begin
  /Mapped Mapped not cursor-state and def
  end
} def

/shape-square-cursor { % - => -
  gsave
  ClientCanvas setcanvas
  newpath
  CharWidth CharHeight scale
  0 0 1 1 rectpath
  cursor-can /Mapped false put
  cursor-can eoreshapcanvas
  cursor-can setcanvas
  0 fillcanvas
  cursor-state { cursor-on } if
  grestore

```



```

} def

/paint-square-cursor { % - => -
  gsave
    cursor-can setcanvas
    PixelMatrix setmatrix
    .5 fillcanvas
    1 setgray
    0 TextVOffset moveto
    TTYFont setfont
    Lines CurY get CurX 1 getinterval show
  grestore
} def

/square-cursor {
  /shape-cursor //shape-square-cursor def
  /paint-cursor //paint-square-cursor def
  /animate-cursor nullproc def
  shape-cursor
} def

/shape-donut-cursor { % - => -
  gsave
    ClientCanvas setcanvas
    cursor-can /Mapped false put
    newpath
    CharWidth CharHeight scale
    .49 .5 .5 0 360 arc
    closepath
    .49 .5 .3 0 360 arc
    cursor-can eoreshapecanvas
    cursor-can setcanvas
    0 fillcanvas
    cursor-state { cursor-on } if
  grestore
} def

/donut-cursor {
  /shape-cursor //shape-donut-cursor def
  /paint-cursor nullproc def
  /animate-cursor nullproc def
  shape-cursor
} def

/shape-error-cursor { % - => -
  gsave
    ClientCanvas setcanvas
    cursor-can /Mapped false put
    newpath
    CharWidth CharHeight scale
    .2 0 moveto
    0 .3 lineto
    .1 .5 lineto
    .2 .5 lineto
    .2 .55 lineto
    .3 .6 lineto
    .4 .55 lineto
    .4 .95 lineto

```

```

    .5 1 lineto
    .6 .95 lineto
    .6 .55 lineto
    .7 .6 lineto
    .8 .55 lineto
    .8 .5 lineto
    .9 .55 lineto
    1 .5 lineto
    1 .3 lineto
    .8 0 lineto
    closepath
    cursor-can eoreshapecanvas
    cursor-can setcanvas
    0 fillcanvas
    cursor-state { cursor-on } if
  grestore
} def

/animate-error-cursor {
  error-count type /integertype eq {
    /error-count
    error-count 1 sub dup 0 le { pop null donut-cursor } if
  def
} if
} def

/error-cursor {
  /shape-cursor //shape-error-cursor def
  /paint-cursor nullproc def
  /animate-cursor //animate-error-cursor def
  shape-cursor
} def

/shape-cursor //shape-donut-cursor def
/paint-cursor nullproc def
/animate-cursor nullproc def

/update-cursor {
  ClientCanvas setcanvas
  CurX CurY char-pos CharHeight div exch CharWidth div exch
  cursor-can setcanvas
  movecanvas
  paint-cursor
} def

/cursor-on {
  gsave
  update-cursor
  cursor-can /Mapped true put
  /cursor-state true def
  grestore
} def

/cursor-off {
  cursor-can /Mapped false put
  /cursor-state false def
} def

```

```

/erase-to-eos {
  ClientCanvas setcanvas
  newpath
  Columns CurX sub CurX CurY char-rect
  CurY 1 add Rows lt {
%      0 CurY 1 add char-pos CharHeight add    % XXX this is stupid
      0 CurY char-pos
      CharWidth Columns mul
      CharHeight Rows CurY sub 1 sub mul neg
      rectpath
  } if
  1 setgray fill
  clear-to-eos
} def

/clear-to-eos {
  clear-to-eol
  CurY 1 add
  dup Columns lt {
    1 Rows 1 sub {
      clear-line
    } for
  } { pop } ifelse
} def

/erase-to-sos {
  ClientCanvas setcanvas
  newpath
  CurY 0 ne {
    0 CurY 1 sub char-pos
    CharWidth Columns mul
    CharHeight CurY mul
    rectpath
  } if
  CurX 0 ne { CurX 0 CurY char-rect } if
  1 setgray fill
  clear-to-sos
} def

/clear-to-sos {
  clear-to-sol
  CurY 0 ne {
    0 1 CurY 1 sub {
      clear-line
    } for
  } if
} def

/erase-to-eol {
  ClientCanvas setcanvas
  newpath
  Columns CurX sub CurX CurY char-rect
  1 setgray fill
  clear-to-eol
} def

/clear-to-eol {
  Lines CurY get

```

```

CurX
Spaces 0 Columns CurX sub getinterval
putinterval
} def

/erase-to-sol {
  ClientCanvas setcanvas
  newpath
  CurX 0 CurY char-rect
  1 setgray fill
  clear-to-sol
} def

/clear-to-sol {
  Lines CurY get
  0
  Spaces 0 CurX getinterval
  putinterval
} def

/erase-line { % line => -
  ClientCanvas setcanvas
  newpath
  Columns 0 2 index char-rect
  1 setgray fill
  clear-line
} def

/clear-line { % line => -
  Lines exch get
  0 Spaces 0 Columns getinterval putinterval
} def

/erase-screen {
  ClientCanvas setcanvas
  clippath 1 setgray fill
  clear-screen
} def

/clear-screen {
  0 1 Rows 1 sub {
    clear-line
  } for
} def

/scroll-region { % first-line last-line+1 region-steps
  dup 0 ne {
    /region-steps exch def
    BottomMargin min
    /region-bottom exch def
    TopMargin max
    /region-first exch def
    /region-height region-bottom region-first sub def

    region-height 0 ne {
      ClientCanvas setcanvas
      region-height region-steps abs gt {
        region-steps 0 lt {

```

```

        region-height region-steps add      % from-height
        region-first      % from-height from-top
    } {
        region-height region-steps sub      % from-height
        region-bottom 1 index sub          % from-height from-top
    } ifelse
    /from-top exch def /from-height exch def

    /Lines [
        Lines aload pop
        Rows dup region-bottom sub roll
        region-height region-steps neg roll
        Rows region-bottom Rows sub roll
    ] def

    0 from-top 1 sub char-pos                % x y
    Columns CharWidth mul                    % x y w
    from-height CharHeight mul neg          % x y w h
    rectpath                                %
    0 region-steps CharHeight mul copyarea
} if

region-steps 0 lt {
    region-first
    dup region-steps sub 1 sub
} {
    region-bottom 1 sub
    dup region-steps sub 1 add
    exch
} ifelse
Rows 1 sub min 0 max /clear-last exch def
Rows 1 sub min 0 max /clear-first exch def

newpath
0 clear-last char-pos
Columns CharWidth mul
clear-first clear-last sub 1 sub neg CharHeight mul
rectpath
1 setgray fill

clear-first 1 clear-last {
    clear-line
} for
} if
} { pop pop pop } ifelse
} def

/default-arg { % default arg# => val
    dup args length ge {
        pop
    } {
        args exch get
        dup null ne { exch } if
        pop
    } ifelse
} def

/num-arg { % char => -

```

```

    args dup length 1 sub                % char [args] len-1
    2 copy get                          % char [args] len-1 arg
    dup null eq { pop 0 } { 10 mul } ifelse
    4 -1 roll                          % [args] len-1 arg*10 char
    48 sub add                          % [args] len-1 arg*10+digit
    put
} def

/qmark { % - => -
    pop
    args ==
    reset-termulator
} def

/semi { % char => -
    pop
    /args [ args aload pop null ] def
} def

/termulate-ansii-sequence { % char => -
    ansii-parse-array 1 index get
    exec
} def

/termulate-escape-prefix { % char => -
    escape-array 1 index get exec
} def

/termulate-ansii-prefix {
    /termulate-char //termulate-ansii-sequence def
    /args [null] def
} def

/escape-prefix {
    /termulate-char //termulate-escape-prefix def
} def

/termulate-ps-string {
    dup 27 eq {
        pop
        reset-termulator
        ps-string 0 ps-string-len getinterval cvx
        { exec } stopped pop
    } {
        ps-string ps-string-len 3 -1 roll put
        /ps-string-len ps-string-len 1 add def
        ps-string-len ps-string length eq {
            /ps-string
                ps-string-len dup add string
                ps-string 1 index copy pop
            def
        } if
    } ifelse
} def

/ps-string-prefix {
    /ps-string 256 string def
    /ps-string-len 0 def

```

```

    /termulate-char //termulate-ps-string def
} def

/printing { % char => -
    pop
    OutputMonitor {
        input-substring input-at
        2 copy
        dup input-len exch sub getinterval
        0 exch {
            32 lt { exit } if
            1 add
        } forall
        2 copy add 1 sub /input-at exch def
        getinterval put-string
    } monitor
} def

/ignore { % char => -
    pop
%   (Ignored %\n) [ 3 -1 roll ] dbgprintf
} def

/badesc { % char => -
    ignore reset-termulator
} def

/doctrl { % char proc => -
    reset-termulator ctrl
} def

/ctrl { % char proc => -
    OutputMonitor {
        exec pop
    } monitor
} def

/ectrl { % char proc => -
    ctrl
    /error-count 20 def error-cursor
} def

/collect-args { % char => -
    args dup length args-needed sub 3 -1 roll put
    /args-needed args-needed 1 sub def
    args-needed 0 eq {
        reset-termulator
        args aload pop function-pending
    } if
} def

/start-supdup { % char function #args => char
    /args-needed exch def
    /args args-needed array def
    /function-pending exch def
    /termulate-char //collect-args def
} def

```

```

/termulate-normal-char { % char => -
    Characters 1 index get exec
} def

/reset-termulator {
    /termulate-char //termulate-normal-char def
} def

% Terminal program itsself

/term {
    ClientCanvas setcanvas
    /infile currentfile def

    KeyProcess null ne {
        KeyProcess killprocess
    } if
    /KeyProcess {
        DoKeyProcess
    } fork def

    /input-string 2048 string def
    /input-substring null def
    /input-len 0 def /input-at 0 def
    /start-printing-at null def

    reset-termulator
    {    this-char
        termulate-char
        /input-at 1 input-at add def
        pause
    } loop
} def

systemdict /bytesavailable known not {
    /bytesavailable {
        pop 1
    } def
} if

/this-char { % - => char
    input-len input-at eq {
        /input-len
        infile bytesavailable dup -1 eq { pop pop exit } if
        input-string length min 1 max
    } def
    /input-substring
    infile
    input-string 0 input-len getinterval
    input-len 1 eq {
        cursor-on
        readstring
        cursor-off
    } {
        readstring
    } ifelse
    not { pop exit } if
} def

```


[illegible]

```

%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

/bell {
  gsave
  ClientCanvas setcanvas
  clippath pathbbox scale pop pop
  0 setgray 5 setrasteropcode
  currenttime BellTime add
  {
    random random random 4 div 0 360 arc
    gsave eofill
    pause dup currenttime le {exit} if
    grestore eofill
  } loop
  pop
  grestore eofill
grestore
} def

/backspace {
  -1 move-cur-x
} def

/tab {
  /CurX
  CurX 8 div floor 1 add 8 mul
  Columns 1 sub min
  def
} def

/new-line {
  Auto-CR { carriage-return } if
  line-feed
} def

/line-feed {
  1 line-feeds
} def

/line-starve {
  1 line-starves
} def

/form-feed {
  home-cur
  erase-screen
} def

/carriage-return {
  /CurX 0 def
} def

/hash-prefix {
  56 eq {
    test-pattern
  } if
  reset-termulator

```

```

} def

/do-hash {
  /termulate-char //hash-prefix def
} def

/test-pattern {
  Columns 0 ne Rows 0 ne and {
    Columns string
    0 1 Columns 1 sub {
      1 index exch 69 put
    } for
    0 1 Rows 1 sub {
      Lines exch get 1 index exch copy pop
    } for
    pop
  } if
  slow-refresh
} def

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%      Escape Sequence Functions
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

/ICH { % char => -
  1 0 default-arg insert-chars
} def

/CUU { % char => -
  /CurY
  CurY
  1 0 default-arg
  1 max sub 0 max
  def
} def

/CUD { % char => -
  /CurY
  CurY
  1 0 default-arg
  1 max add Rows 1 sub min
  def
} def

/CUF { % char => -
%   1 0 default-arg 1 max move-cur-x
%   1 0 default-arg 1 max
  CurX add Columns 1 sub min /CurX exch def
} def

/CUB { % char => -
%   1 0 default-arg neg -1 min move-cur-x
%   1 0 default-arg neg -1 min
  CurX add 0 max /CurX exch def
} def

```

```

/CNL { % char => -
    0 Rows 1 sub CurY 1 0 default-arg add min
    set-cur
} def

/HVP { % char => -
    1 1 default-arg 1 sub
    1 0 default-arg 1 sub
    set-cur
} def

/CUP //HVP def

/ED { % char => -
    0 0 default-arg {
        0 { erase-to-eos }
        1 { erase-to-sos }
        2 { erase-screen }
    } case
} def

/EL { % char => -
    0 0 default-arg {
        0 { erase-to-eol }
        1 { erase-to-sol }
        2 { CurY erase-line }
    } case
} def

/IL { % char => -
    CurY
    1 0 default-arg
    insert-lines
} def

/DL { % char => -
    CurY
    1 0 default-arg
    delete-lines
} def

/DCH { % char => -
    1 0 default-arg delete-chars
} def

/SGR { % char => -
    /graphics-rendition 0 0 default-arg def
} def

/SUNBOW { % char => -
    /reverse-video false def
} def

/SUNWOB { % char => -
    /reverse-video true def
} def

```

```

/SCRL { % char => -
  1 0 default-arg dup 0 eq { pop 1 } if
  1 sub Rows 1 sub min
  /TopMargin exch def
  Rows 1 default-arg dup 0 eq { pop 1 } if
  Rows min TopMargin 1 add max
  /BottomMargin exch def
  0 0 set-cur
} def

```

```

/SUNRESET { % char => -
  % XXXX
} def

```

```

/save-cursor-pos { %
  /SaveCurX CurX def
  /SaveCurY CurY def
} def

```

```

/restore-cursor-pos {
  SaveCurX SaveCurY set-cur
} def

```

```

/NEL {
  carriage-return line-feed
} def

```

```

/zPSH {
  % Write me
} def

```

```

/zPOP {
  % Write me
} def

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% Character dispatch table
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

/Characters [
  % @ABC DEFG
%   //ignore //ignore //ignore //ignore
  //ignore { //stack ctrl} //ignore //ignore
  //ignore //ignore //ignore { //bell ctrl }
  % HIJK LMNO
  { //backspace ctrl} { //tab ctrl} { //new-line ctrl} { //line-starve ctrl}
  { //form-feed ctrl} { //carriage-return ctrl} //ignore //ignore
  % PQRS TUVW
  //ignore //ignore //ignore //ignore
  //ignore //ignore //ignore //ignore
  % XYZ[ \]^_
  //ignore //ignore //ignore { //escape-prefix ctrl}
  //ignore //ignore //ignore { //ps-string-prefix ctrl}
  % printing
  96 { //printing } repeat
] def

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% Parse Dictionary
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

/escape-dict 100 dict begin
  ([) 0 get
    { //termulate-ansii-prefix ctrl } def
  (#) 0 get
    { //do-hash doctrl } def
  (7) 0 get
    { //save-cursor-pos doctrl } def
  (8) 0 get
    { //restore-cursor-pos doctrl } def
  (D) 0 get
    { //line-feed doctrl } def
  (M) 0 get
    { //line-starve doctrl } def
  (E) 0 get
    { //NEL doctrl } def
  (H) 0 get
    { //HTS doctrl } def
  (c) 0 get
    { //SUNRESET doctrl } def
currentdict end def

```

```

/escape-array [
  0 1 255 {
    escape-dict 1 index known {
      escape-dict exch get
    } {
      pop
      //badesc
    } ifelse
  } for
] def

```

```

/escape-dict null def % lazy for now

```

```

/ansii-parse-array [
% Control
% @ABC DEFG
//ignore //ignore //ignore //ignore
//ignore //ignore //ignore { //bell ectrl }
% HIJK LMNO
{//backspace ectrl} {//tab ectrl}
{//new-line ectrl} {//line-starve ectrl}
{//form-feed ectrl} {//carriage-return ectrl}
//ignore //ignore
% PQRS TUVW
//ignore //ignore //ignore //ignore
//ignore //ignore //ignore //ignore
% XYZ[ \]^_
//ignore //ignore //ignore {//escape-prefix ectrl}
{{}} doctrl} //ignore //ignore {//ps-string-prefix ectrl}
% Printing

```

```

% <space>!"#$%&'
//badesc //badesc //badesc //badesc //badesc //badesc //badesc //badesc
% ()*+,-./
//badesc //badesc //badesc //badesc //badesc //badesc //badesc //badesc
% 0123456789
10 { //num-arg } repeat
% :;<=>?
//badesc //semi //badesc //badesc //badesc //badesc
% @ABC DEFG
{//ICH doctrl} {//CUU doctrl} {//CUD doctrl} {//CUF doctrl}
{//CUB doctrl} {//CNL doctrl} //badesc //badesc
% HIJK LMNO
{//CUP doctrl} //badesc {//ED doctrl} {//EL doctrl}
{//IL doctrl} {//DL doctrl} //badesc //badesc
% PQRS TUVW
{//DCH doctrl} //badesc //badesc {//SUNRESET doctrl}
//badesc //badesc //badesc {//CTC doctrl}
% XYZ[ \]^_
{//ECH doctrl} //badesc //badesc //badesc
//badesc //badesc //badesc //badesc
% `abc defg
//badesc {//CUF doctrl} //badesc //badesc
//badesc {//CUD doctrl} {//HVP doctrl} {//TBC doctrl}
% hijk lmno
//badesc //badesc //badesc //badesc
//badesc {//SGR doctrl} //badesc //badesc
% pqrs tuvw
{//SUNBOW doctrl} {//SUNWOB doctrl} {//SCRL doctrl} {//zPSH doctrl}
{//zPOP doctrl} //badesc //badesc //badesc
% xyz{ |}~<rub>
//badesc //badesc //badesc //badesc
//badesc //badesc //badesc //badesc
] def

```

```

classend def

```

```

% This is executed when psexec (the C program) connects to the server
% and sends the string "psexec\n". It opens a terminal window and
% starts the server listening for input. A program is then exec'ed
% with standard in, out, and error hooked up to the input and output
% of the terminal emulator in the server.

```

```

/strip-domain { % host.domain => host
  (.) search {
    exch pop exch pop } if
} def

```

```

/psexec {
% 20 dict begin
  /myterm framebuffer /new TTY send def
  count 0 ne {
    dup type /arraytype eq {
      cvx /doit myterm send
    } if
  } if
  myterm /FrameWidth get 0 eq {
    /reshapefromuser myterm send
  } if

```

```
        /map myterm send
        /term myterm send
% end
} def

end
```